chapter-01-01

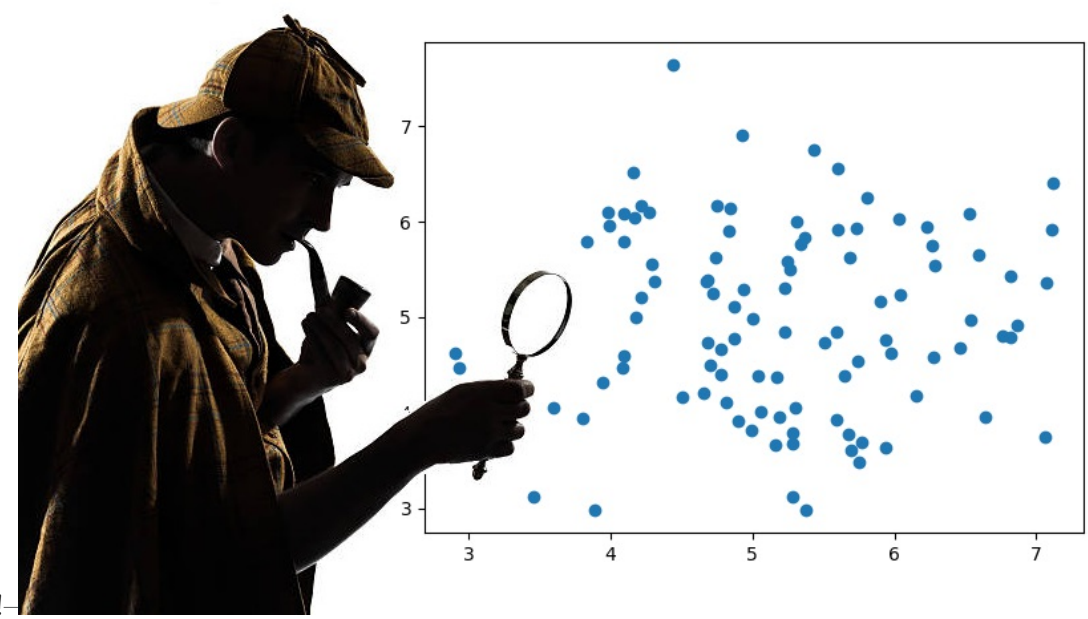
March 14, 2025

Run in Google Colab

1 A Short Course on Advanced Unsupervised Learning

1.1 About these Lessons

- Welcome to our **journey through the world of advanced unsupervised machine learning models**!
- As we embark on this adventure together, I'm going to introduce you to a realm of algorithms and techniques that uncover hidden patterns and structures in data without the need for explicit labeling.
- Unsupervised learning is a bit like being a detective, where you don't have a straightforward case to solve but rather clues and pieces of evidence to piece together into a coherent story.



<!–

Introduction to Machine Learning

- We'll start this journey with the basics, exploring how unsupervised learning differs from its supervised counterpart and the types of problems it's best suited for.

- We'll give also a short introduction to general Machine Learning Problem in order to make this part self-consistent.

Before Leaving...

- As we journey through these topics, remember: don't worry if you don't understand all the topics we are going to talk about in these days!
- The aim of the seminar is to give an overview of the main techniques used in a field that is having a growing application interest. No one expects you to become machine learning experts in three lessons.
- The purpose of these lessons is only to give a general overview and spark curiosity
- You can find all the teaching matherials both on **Virtuale** and on my GitHub repository **https://github.com/polyhedron-gdl**
- Don't worry if you are not an experienced programmer, the Python code is fully commented and Python is a very simple language to understand
- The test, regarding this part of the course, will consist of a series of simple multiple choice questions
- So don't worry, just relax and enjoy the ride …

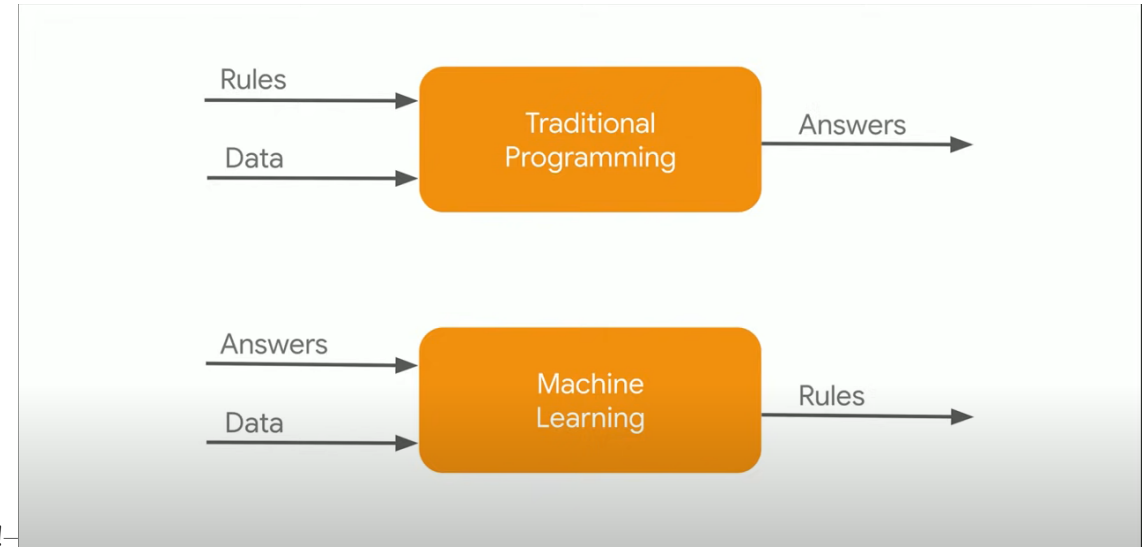
1.2 Introduction to Machine Learning

What is Machine Learning

Machine learning (ML) is a branch of artificial intelligence (AI) focused on building systems that learn from data. At its core, machine learning enables computers to identify patterns and make decisions with minimal human intervention. The foundational principle of machine learning is to develop algorithms that can process large amounts of data, learn from this data, and then apply what they have learned to make informed decisions or predictions about new, unseen data.

The Paradigm of Machine Learning

- In traditional programming, humans provide the rules (algorithms) and the data, and the program processes this information to produce answers.
- In the machine learning process, instead of being programmed with explicit rules, the system is fed with data and the answers (often known as the 'ground truth').
- It then learns the rules by identifying patterns in the data to arrive at the given answers.



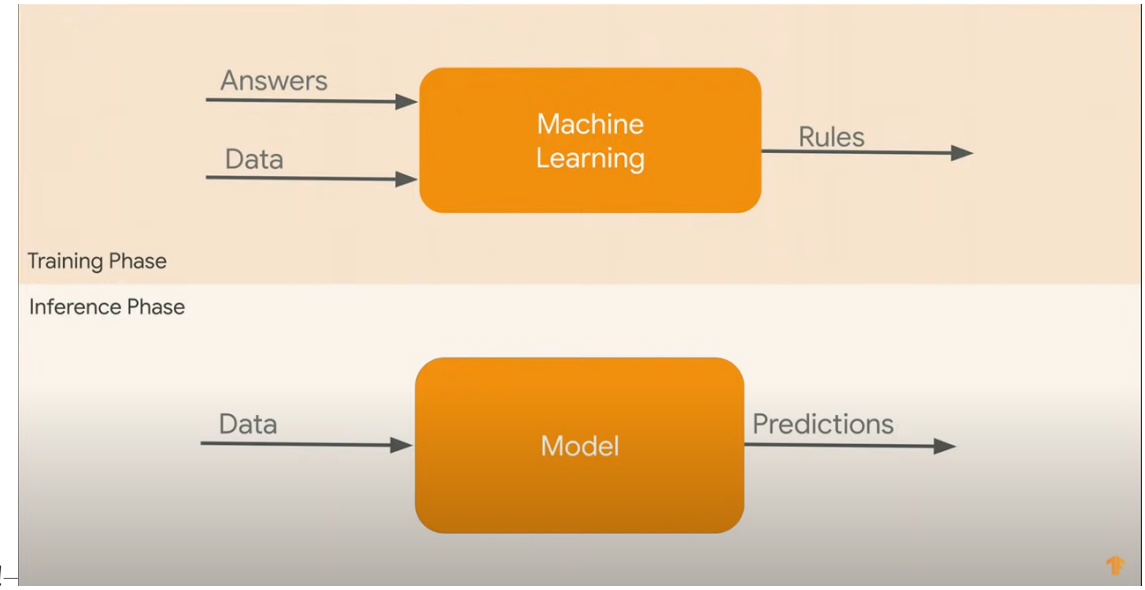
<!–

Key Steps

- The process of machine learning involves several key steps:
  - data collection,
  - data preprocessing,
  - model selection,
  - training,
  - evaluation,
  - deployment (eventually).
- Data collection encompasses gathering the raw data needed for the learning process. This data can come from a wide range of sources, including but not limited to, databases, sensors, and user interactions.
- Data preprocessing involves cleaning and organizing this data into a format that can be effectively used by machine learning models. This step often includes handling missing values, normalizing data, and encoding categorical variables.

Training

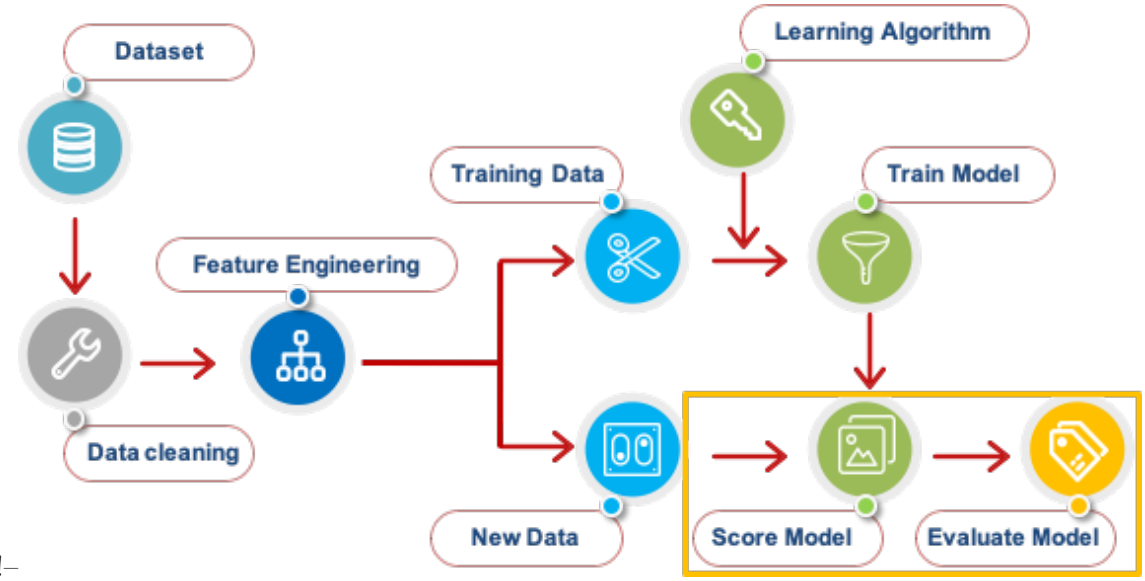
- Training the model involves feeding the preprocessed data into the model, allowing it to learn and adjust its parameters.
- This is typically done through a process called optimization, where the model makes predictions on the training data and adjusts its parameters to minimize the difference between its predictions and the actual outcomes.



<!–

Evaluation

- Evaluation is a critical step where the trained model is tested on a separate dataset to assess its performance.
- This helps to ensure that the model can generalize well to new, unseen data. Common metrics used for evaluation include accuracy, precision, recall, and the F1 score for classification problems, and mean squared error or mean absolute error for regression problems.



<!–

Model Selection

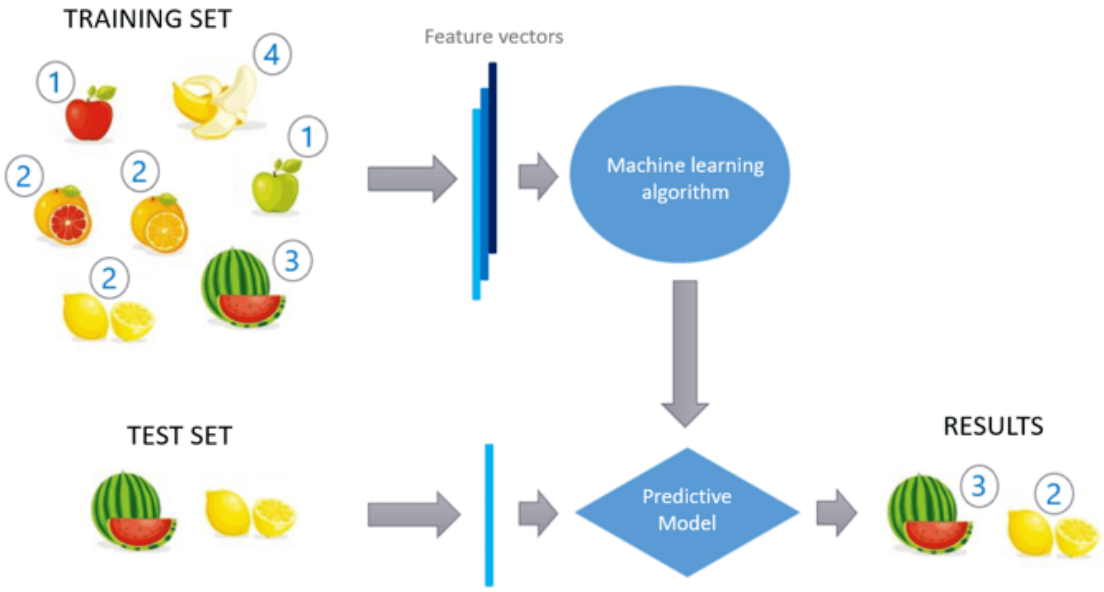
Model selection is the process of choosing the appropriate algorithm to solve a specific problem. There are many types of machine learning models, including - supervised learning models, where the algorithm learns from a labeled dataset; - unsupervised learning models, where the algorithm learns patterns from unlabeled data; - reinforcement learning models, where an agent learns to make decisions by interacting with an environment.

Let's describe in more details the difference among these models…

1.3 The three different types of machine learning

1.3.1 Supervised Learning

- The main goal in supervised learning is to learn a model from labeled training data that allows us to make predictions about unseen or future data. Here, the term "supervised" refers to a set of **training** examples (data inputs) where the desired output signals (**labels**) are already known.

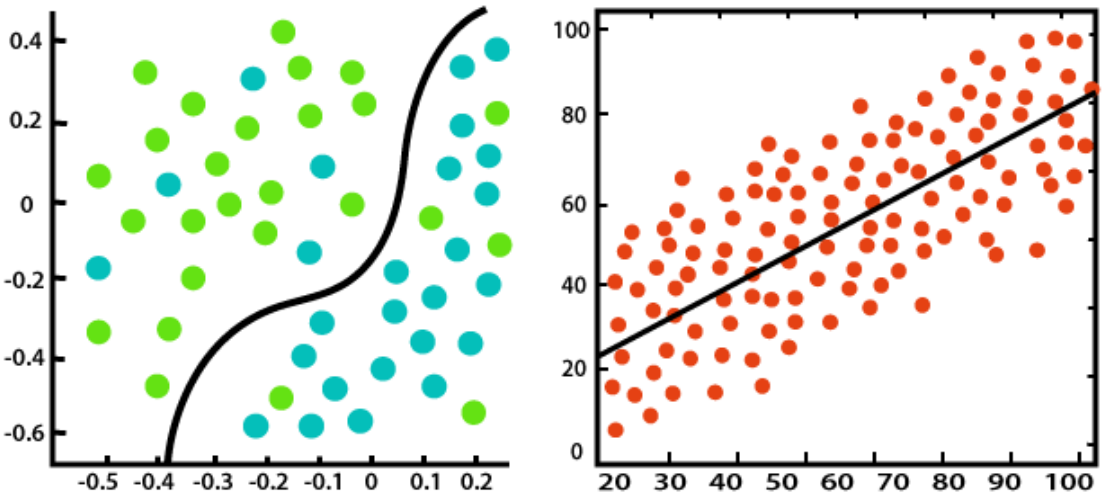


<!–

The two most important types of Supervised Models

- A supervised learning task with discrete class labels, such as in the previous example, is also called a **classification task**.
- A second type of supervised learning is the prediction of continuous outcomes, which is also called **regression analysis**. In regression analysis, we are given a number of predictor (explanatory) variables and a continuous response variable (outcome), and we try to find a relationship between those variables that allows us to predict an outcome.

In the field of machine learning, the predictor variables are commonly called ***features***, and the response variables are usually referred to as ***target variables*** or ***labels***. In other words, "features" are the pieces of data that you use to describe each observation in a way that a computer can understand. Imagine you want to teach a machine to distinguish between apples and oranges. Features could be things like the color, diameter, and weight of the fruit. Each fruit (apple or orange) is an observation, and the features are the details about the fruit that you're using to help the machine learn to tell them apart. In other words, features are the inputs that you give to a machine learning algorithm to help it make decisions or predictions.



<!–

1.3.2 Unsupervised Learning

- In supervised learning, we know the right answer beforehand when we train a model.
- In **unsupervised learning**, however, we are dealing with ***unlabeled data*** or data of unknown structure.
- Using unsupervised learning techniques, we are able to explore the structure of our data to extract meaningful information without the guidance of a known outcome variable or reward function.

Clustering

- Clustering is an exploratory data analysis technique that allows us to organize a pile of information into meaningful subgroups (clusters) *without having any prior knowledge of their group memberships.*
- Each cluster that arises during the analysis defines a group of objects that share a certain degree of **similarity** but are more dissimilar to objects in other clusters, which is why clustering is also sometimes called unsupervised classification. Clustering is a great technique for structuring information and deriving meaningful relationships from data.

For example, it allows marketers to discover customer groups based on their interests, in order to develop distinct marketing programs.

Clustering IS NOT Classification

1.3.3 Reinforcement Learning

- Another type of machine learning is **reinforcement learning**.
- In reinforcement learning, the goal is to develop a system (***agent***) that improves its performance based on interactions with the environment.

- Since the information about the current state of the environment typically also includes a so-called **reward signal**, we can think of reinforcement learning as a field related to supervised learning.
- However, in reinforcement learning, this feedback is not the correct ground truth label or value, but a measure of how well the action was measured by a reward function. Through its interaction with the environment, an agent can then use reinforcement learning to learn a series of actions that maximizes this reward via an exploratory trial-and-error approach or deliberative planning. A popular example of reinforcement learning is a chess engine. Here, the agent decides upon a series of moves depending on the state of the board (the environment), and the reward can be defined as win or lose at the end of the game.

1.4 Features and Labels

- The data for supervised learning contains what are referred to as **features** and **labels**.
- The **labels** are the values of the target that is to be predicted.
- The **features** are the variables from which the predictions are to be made.

For example when predicting the price of a house the **features** could be the square meters of living space, the number of bedrooms, the number of bathrooms, the size of the garage and so on. The **label** would be the house price.

Features and Vector Spaces

- The connection between features in machine learning and coordinates in a vector space is quite fundamental.
- In machine learning, when we describe an observation using features, we are effectively translating that observation into a point in a multi-dimensional space, where each dimension represents a feature.
- In machine learning, every observation (like a house in our example) can be represented as a vector. The features of the observation are the **coordinates of the vector** in this high-dimensional space. Each feature corresponds to one axis of the space. So, if we have three features, each observation is a point in a 3-dimensional vector space. If we have ten features, it's a 10-dimensional space.
- When you plot these vectors (points) in the feature space, the patterns that emerge can be used by machine learning models to make predictions. For instance, points that are close together are likely to represent similar observations, and a machine learning model might predict similar outcomes for them.

The main difference between supervised and unsupervised learning: Labeled data

- Risking of being repetitive, I want to stress again that the main distinction between the two approaches is the use of labeled datasets. To put it simply, supervised learning uses labeled input and output data, while an unsupervised learning algorithm does not.
- Unsupervised learning models, in contrast, work on their own to discover the inherent structure of unlabeled data. Note that they still require some human intervention for validating output variables.

1.5 Basic of Unsupervised Learning

Pattern Detection

- As we have seen, at its core, unsupervised learning involves the analysis of data sets without predefined or known outcomes.
- The algorithms seek to identify patterns or groupings from the input data without any guidance or supervision.
- This form of learning is crucial when the task at hand does not include prior knowledge, or when it is impractical to obtain labeled data, which is often expensive and time-consuming.
- Unsupervised learning, unlike its counterpart supervised learning, operates on **data without labeled responses**.
- The primary goal is **to unearth hidden patterns, intrinsic structures, or useful representations** from such unlabeled data.

Key Techniques in Unsupervised Learning

- **Clustering**: Clustering is perhaps the most well-known unsupervised learning technique. It involves grouping a set of objects in such a way that objects in the same group (called a cluster) are more similar to each other than to those in other groups. Common clustering algorithms include K-Means, hierarchical clustering, and DBSCAN.
- **Dimensionality Reduction**: This technique is about reducing the number of random variables under consideration and can be divided into feature selection and feature extraction. Methods like Principal Component Analysis (PCA), t-Distributed Stochastic Neighbor Embedding (t-SNE), and autoencoders are widely used for dimensionality reduction.
- **Association Rule Learning**: This technique is used to discover interesting relations between variables in large databases. It's commonly used in market basket analysis where it reveals how items purchased by customers are related. The Apriori algorithm is a classic example of association rule learning.
- **Anomaly Detection**: Anomaly detection involves identifying unusual patterns that do not conform to expected behavior. It is widely used in fraud detection, system health monitoring, and outlier detection in data cleaning and preprocessing.

Applications of Unsupervised Learning

Unsupervised learning techniques are applied in numerous fields due to their ability to discover hidden patterns in unlabeled data.

- In **marketing**, clustering helps in customer segmentation by identifying groups of customers with similar behaviors or preferences.
- In **genomics**, it assists in understanding genetic structures and variations by clustering similar genetic patterns.
- **Finance** sector employs anomaly detection for fraudulent transaction identification.
- In **image processing**, unsupervised learning helps in image compression and segmentation.
- **Natural Language Processing** (NLP) utilizes unsupervised learning for topic modeling and word clustering.

Challenges and Considerations in Unsupervised Learning

While unsupervised learning is powerful, it comes with its set of challenges:

- **Interpretability**: The outcomes of unsupervised learning are often difficult to interpret. Since there are no predefined labels, the meaning and significance of the results can be ambiguous and require domain expertise for interpretation.
- **Evaluation Metrics**: Evaluating the performance of unsupervised learning models is challenging since there is no ground truth to compare against. Metrics such as silhouette score or Davies-Bouldin index are used in clustering, but they don't always provide a clear indication of model performance.
- **Data Quality**: The quality of outcomes heavily depends on the quality of input data. Noisy, incomplete, or inconsistent data can lead to misleading patterns and results.

1.6 Learning Tools

1.6.1 Using Python for machine learning

- Python is one of the most popular programming languages for data science and thanks to its very active developer and open source community, a large number of useful libraries for scientific computing and machine learning have been developed.
- Although the performance of interpreted languages, such as Python, for computation-intensive tasks is inferior to lower-level programming languages, extension libraries such as **NumPy**, **Matplotlib** and **Pandas**, among the others, have been developed that build upon lower-layer Fortran and C implementations for fast vectorized operations on multidimensional arrays.
- For machine learning programming tasks, we will mostly refer to the **scikit-learn** library, which is currently one of the most popular and accessible open source machine learning libraries.
- In the later chapters, when we focus on deep learning, we will use the latest version of the **Keras** library, which specializes in training so-called deep neural network models very efficiently.

1.6.2 Installing Python and Packages

To set up your python environment, you'll first need to have a python on your machine. There are various python distributions available and we have chosen one that works very well for data science: **Anaconda**. Anaconda comes with its own Python distribution which will be installed along with it.

Data Science often requires you to work with a lot of scientific packages like scipy and numpy, data manipulation packages like pandas and IDEs and interactive Jupyter Notebook.Now, you don't need to worry about any python package most of them come pre-installed and if you want to install a new package, you can do that simply by using conda or via the pip installer program, which has been part of the Python Standard Library since Python 3.3. More information about pip can be found here. After we have successfully installed Python, we can execute pip from the terminal to install additional Python packages:

pip install SomePackage

Already installed packages can be updated via the –upgrade flag:

pip install SomePackage –upgrade

To download an Anaconda distribution, you can use the official download page and you can select your platform and then choose the installer. For this, you can choose which version you want and whether 32-bit or 64-bit.

To test your installation, on Windows, click on Start and then Anaconda Navigator in the program list (or search for Anaconda in the search bar and select Anaconda Navigator). On a Mac, open up the finder, and in the Applications folder, double click on Anaconda-Navigator.

Package Managers

Anaconda will give you two package managers- **pip** and **conda**. When some packages aren't available with conda, you can use pip to install them. Note that using pip to install packages also available to conda may cause an installation error.

Jupyter Notebook

- Jupyter Notebook is a graphical user interface (GUI) for running code interactively and interleaving it with text documentation and figures. Due to its versatility and ease of use, it has become one of the most popular tools in data science.
- A notebook is a document like this one! A notebook integrates code and its output into a single document that combines visualizations, narrative text, mathematical equations, and other rich media.
- In other words: it's a single document where you can run code, display the output, and also add explanations, formulas, charts, and make your work more transparent, understandable, repeatable, and shareable. As part of the open source Project Jupyter, Jupyter Notebooks are completely free. You can download the software on its own, or as part of the Anaconda data science toolkit.
- For more information about the general Jupyter Notebook GUI, please view the official documentation here.

We highly recommend Adam Rule et al.'s article "Ten simple rules for writing and sharing computational analyses in Jupyter Notebooks on using Jupyter Notebook effectively" in scientific research projects, which is freely available here

1.6.3 Google Colab

- Google Colaboratory, commonly known as Google Colab, is a free cloud service hosted by Google to encourage machine learning and artificial intelligence research, making it accessible to a broader audience.
- Colab provides a platform that enables users to write and execute Python code through a web browser without requiring any setup. It provides Jupyter Notebook instances that run on the cloud; the notebooks can be saved on Google Drive or GitHub.
- It is particularly favored for its ability to offer free access to computing resources, including GPUs and TPUs, which can significantly speed up computation times for data analysis and

model training, making advanced machine learning projects feasible for those without access to high-end hardware.

- Accessing Google Colab is very straightforward. You can visit https://colab.research.google.com, which automatically takes you to a prompt window where you can see your existing Jupyter notebooks.

Integration with Google Drive

- One of the key features of Google Colab is its integration with Google Drive, allowing users to easily share their work, collaborate in real-time with others, and access their notebooks from anywhere.
- This feature fosters a collaborative environment, making it easier for users to work together on projects, share insights, and learn from each other.

Colab works with Jupyter Notebook

- Colab notebooks support various popular machine learning libraries, such as TensorFlow, PyTorch, Keras, and OpenCV, enabling users to experiment with deep learning and other advanced algorithms.
- These notebooks are interactive, allowing the inclusion of rich media, documentation, and even interactive widgets, which enhances the learning and development process.

Although it is not essential to work in a colab environment (all the course notebooks are in fact designed to be able to run without problems locally on your pc), it is useful to know some basic elements of the interaction with colab. In particular, in the cells below you will find two examples for the use of external files. In the first case it is shown how to load a text file from your local PC into the google virtual machine. The second example relates to the opposite operation: let's create a simple pandas dataframe into the colab environment and export it in csv format to the local machine.

How Upload a File on Google Colab

[6]:
```
if 'google.colab' in str(get_ipython()):
    from google.colab import files
    uploaded = files.upload()
    path = ''
else:
    path = './data/'
```

[7]:
```
with open(path + "countryriskdata.csv", "r") as f:
    file = f.read()

file[:100]
```

[7]:
```
'Country,Abbrev,Corruption,Peace,Legal,GDP
Growth\nAlbania,AL,39,1.867,3.822,3.403\nAlgeria,DZ,34,2.213'
```

How Download a File on Google Colab

[8]:
```
import pandas as pd

cars = {'Brand': ['Honda Civic','Toyota Corolla','Ford Focus','Audi A4'],
        'Price': [22000,25000,27000,35000]
        }

df = pd.DataFrame(cars, columns= ['Brand', 'Price'])
```

[9]:
```
if 'google.colab' in str(get_ipython()):
    # if we run in google environment first we save in virtual machine...
    df.to_csv ('export_dataframe.csv', index = False, header=True)
    # ...then we download to local machine
    from google.colab import files
    files.download("export_dataframe.csv")
else:
    # if we are working in local we save directly with the usual method
    df.to_csv ('./data/export_dataframe.csv', index = False, header=True)
```

1.6.4 Data Science Python Libraries

As we delve into the multifaceted world of machine learning, a number of libraries stand out for their robustness and versatility. Among these, Scikit-learn, SciPy, Keras are cornerstones in the Python ecosystem for data science and provide a suite of tools that are indispensable for machine learning practitioners.

- It is a scientific computing library that provides fundamental functionalities for mathematics, science, and engineering. It extends the capabilities of NumPy with additional modules for optimization, linear algebra, integration, interpolation, special functions, FFT, signal and image processing, ODE solvers, and more.
- While SciPy is not exclusively a machine learning library, it forms the backbone of many higher-level machine learning operations that require scientific computations. Its modules are meticulously optimized for performance and are relied upon by researchers and developers for technical and scientific computing tasks that demand high precision and efficiency.
- Commonly referred to as sklearn, is a specialized library that offers a wide array of machine learning algorithms and tools.
- It is built on top of libraries such as NumPy, SciPy, and matplotlib, which are workhorses for numerical computing and data visualization in Python. Scikit-learn simplifies complex processes, allowing for the easy implementation of many machine learning techniques.
- It encompasses algorithms for classification, regression, clustering, and dimensionality reduction, as well as utilities for model evaluation, data transformation, and data splitting. The library's consistency in API design makes it highly accessible for beginners, yet it remains powerful enough for seasoned practitioners to implement state-of-the-art machine learning models with only a few lines of code.
- Keras is an open-source neural network library designed for ease of use, flexibility, and modularity, Keras acts as an interface for the TensorFlow library, allowing for the construction, training, and deployment of machine learning models with high-level building blocks. It is

particularly favored for its user-friendly API, which enables rapid prototyping of deep learning models.

- With Keras, developers can build complex machine learning models with minimal lines of code. It supports a wide range of network architectures, including fully connected, convolutional, and recurrent neural networks, as well as combinations of the two. Keras simplifies the process of building and training models by providing a set of pre-defined layers, optimizers, and utility functions. This makes it accessible to both beginners in machine learning and experienced researchers by abstracting away much of the complexity involved in developing deep learning models.

References and Additional Infos

- For those eager to explore the intricacies of Scikit-learn and SciPy, there are a wealth of resources available that range from official documentation to comprehensive textbooks and online courses.
- To delve into **Scikit-learn**, the library's official documentation (https://scikit-learn.org/stable/documentation.html) is the definitive reference, offering detailed guides and tutorials on every aspect of the library. It includes user guides for different machine learning algorithms, information on model selection and evaluation, and practical examples to get your hands dirty. For a more structured learning experience, ***"Hands-On Machine Learning with Scikit-Learn, Keras, and TensorFlow"*** by Aurélien Géron provides a deep dive into using Scikit-learn for practical machine learning. This book is well-regarded for its clear explanations and hands-on approach.
- When it comes to **SciPy**, the official documentation (https://docs.scipy.org/doc/scipy/reference/) is again an excellent starting point. It provides detailed documentation of all modules and functions within the library. For a broader understanding, ***"Python for Data Analysis"*** by Wes McKinney offers insights into using SciPy alongside pandas, NumPy, and other data analysis tools. For those who prefer a more interactive approach, platforms like Coursera, edX, and Udemy offer courses on scientific computing with Python that include modules on SciPy.

In addition to these resources, communities such as Stack Overflow and GitHub provide forums where one can ask questions, share knowledge, and collaborate on projects. Journals such as the Journal of Machine Learning Research (JMLR) and conferences like SciPy and PyCon also publish papers and talks on the latest developments and applications of these libraries. These resources collectively provide a comprehensive ecosystem for learners to deepen their understanding and expertise in using Scikit-learn and SciPy for machine learning and scientific computing.

1.7 Emerging Trends and Future Directions

Unsupervised learning is an area ripe for innovation and growth. Recent trends include the integration of unsupervised learning with deep learning techniques, such as deep neural networks and autoencoders. These approaches have shown promising results in complex tasks like feature learning, representation learning, and generative models.

Another exciting development is the use of unsupervised learning in reinforcement learning and transfer learning, where it helps in feature discovery and efficient learning in environments with sparse or no labels.

1.8 Conclusion

- Unsupervised learning is a dynamic and expansive field in machine learning. Its ability to work with unlabeled data makes it incredibly versatile and valuable across various domains. As data continues to grow in size and complexity, the role of unsupervised learning in extracting meaningful information and discovering hidden patterns becomes increasingly important.
- While it poses unique challenges in terms of interpretation and evaluation, advancements in algorithms and computational power continue to push the boundaries, making unsupervised learning an exciting field to watch in the coming years.
- The future of unsupervised learning, intertwined with developments in artificial intelligence, holds immense potential for innovation and discovery, making it a key pillar in the quest to harness the power of data.

1.9 References and Credits

[ ]: